A Appendix

This section provides additional implementation details, background information, and results. Therefore, it is started by introducing the distribution strategy used to parallelize the training of the TD3 agents. Thereafter, the adaptions to the simulation setup used to generate the presented results are introduced. Finally, the missing result plots of the exemplary trajectories shown in the main paper – c.f., Figure 5 and 6 – are presented. Note that, videos, images, and more discussion can be found on our project webpage[1].

A.1 Distribution Strategy

While OpenAI-ES offers straightforward parallelization to a huge number of workers by default, TD3 was introduced as a serial approach. Nonetheless, as TD3 is an off-policy approach, it can be trained on data from parallel workers as well. Barth-Maron et al. [33] introduced a distributed version of DDPG, which runs multiple actors in parallel to generate experience that is added to a joint replay buffer. Then a single critic can be trained on data from the joint replay buffer. We also distribute training in TD3. However, unlike Barth-Maron et al. [33], we only distribute the simulation environment to different workers and run it with different seeds while using a single actor instead. This distribution strategy allows more efficient training on consumer PCs as it reduces communication bandwidth – only states and actions need to be communicated instead of actor parameters – and runs the single actor on the GPU to predict the actions of all workers at once. In contrast, we implemented OpenAI-ES similarly to the original authors, as it relies on the usage of different actors.

A.2 Simulation Setup

RLBench is based on PyRep and CoppeliaSim, and provides the reaching and pick-and-lift tasks. However, some adaptions had to be made to the tasks and RLBench itself. At the time of writing, RLBench uses different time horizons for sole arm actions and arm actions with gripper usage. While a sole arm action is executed in a single time step, once a gripper action is triggered, the simulation runs in a loop until the gripper has changed its state while continuously repeating the arm action. This idiosyncrasy was learned by the agent and led to inferior performance. Thus we separated arm and gripper actions: once a gripper action is triggered, arm actions are stalled. This is also beneficial for real-world application, as this relationship between arm and gripper actions allows scaling the arm actions almost independently of the gripper actuation. Finally, the target object in the pick-and-lift task was not rotated during training to lower the task's complexity.

A.3 Results

Pick-and-Lift Task with an Inclined Reference Configuration When using the loss that minimizes the distance to a reference configuration – c.f., Equation (9) –, any reference configuration can be passed. In Figure 5, an inclined reference configuration was passed to learn to grasp from the right. Figure 7 presents the respective validation results of the TD3 agent on the pick-and-lift task with an inclined reference configuration during training. As can be seen, our approach reaches similar performance to the agent without redundancy resolution on the main objective (even slightly outperforming the agent without redundancy resolution), while allowing to embed secondary objectives – here minimizing the distance to an inclined reference configuration.

Reaching Task with Collision Avoidance When using the loss shown in Equation (10) as the secondary objective, the agent is biased towards actions that maximize the distance between the robot links and obstacles, as shown in the exemplary episode presented in Figure 6. Figure 8 presents the respective validation results of the TD3 agent on the reaching task during training. As can be seen, our method again reaches similar performance to the agent without redundancy resolution on the main objective, while allowing to embed secondary objectives – here collision avoidance.

[1] https://sites.google.com/view/redundant-action-bias

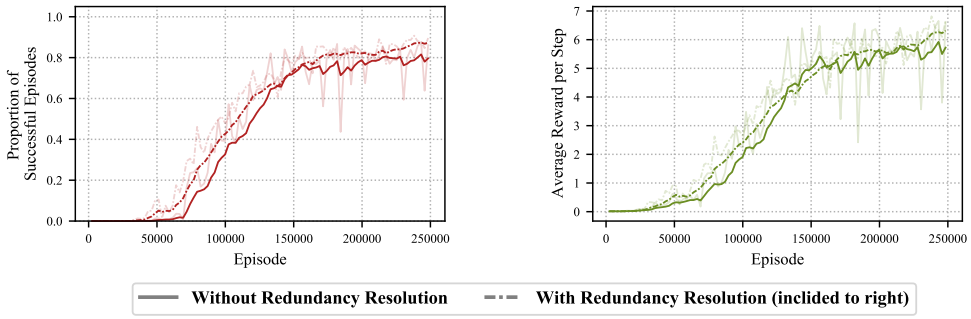

Figure 7: Validation results of TD3 on a pick-and-lift task with an inclined reference position during redundancy resolution in order to learn grasping from the right. The abscissa shows training episodes. Every 2500 training episodes, the training was paused and the agents were evaluated on 1000 random validation episodes (i.e., no exploration). Then, the means were taken to plot a single point of the reward and the loss.

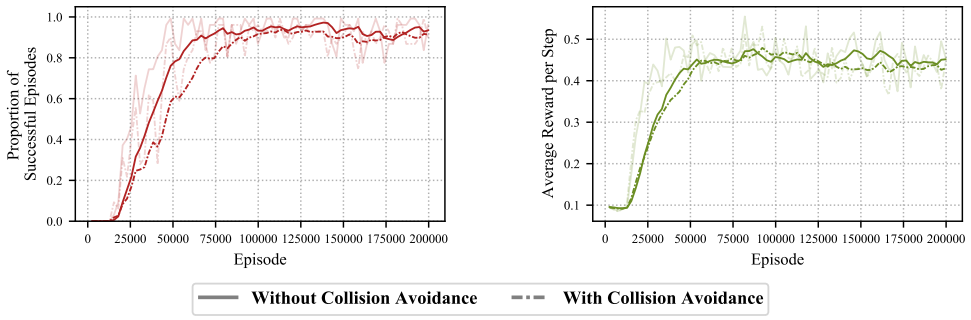

Figure 8: Validation results of TD3 on a reaching task with and without redundancy resolution for collision avoidance. The abscissa shows training episodes. Every 2500 training episodes, the training was paused and the agents were evaluated on 1000 random validation episodes (i.e., no exploration). Then, the means were taken to plot a single point of the reward and the loss.